In [1]:
```
# import zipfile
# zip_file = 'fake-news.zip'
# destination_folder = './fake_news/unzipped'
# with zipfile.ZipFile(zip_file, 'r') as zip_ref:
#     zip_ref.extractall(destination_folder)
# print('The contents of the zip file have been extracted succefully')
```

# About the dataset:

- id - unique identity for news article
- title - the title of the news article
- author - author of the news article
- text - the test of the article
- label - to mark the real vs fake article

# 0 = real news

# 1 = fake news

In [2]:
```
import pandas as pd
```

In [3]:
```
df = pd.read_csv('fake_news/train.csv')
```

In [4]:
```
df
```

Out[4]:

| | id | title | author | text | label |
|---|---|---|---|---|---|
| **0** | 0 | House Dem Aide: We Didn’t Even See Comey’s Let... | Darrell Lucus | House Dem Aide: We Didn’t Even See Comey’s Let... | 1 |
| **1** | 1 | FLYNN: Hillary Clinton, Big Woman on Campus - ... | Daniel J. Flynn | Ever get the feeling your life circles the rou... | 0 |
| **2** | 2 | Why the Truth Might Get You Fired | Consortiumnews.com | Why the Truth Might Get You Fired October 29, ... | 1 |
| **3** | 3 | 15 Civilians Killed In Single US Airstrike Hav... | Jessica Purkiss | Videos 15 Civilians Killed In Single US Airstr... | 1 |
| **4** | 4 | Iranian woman jailed for fictional unpublished... | Howard Portnoy | Print \nAn Iranian woman has been sentenced to... | 1 |
| **...** | ... | ... | ... | ... | ... |
| **20795** | 20795 | Rapper T.I.: Trump a ’Poster Child For White S... | Jerome Hudson | Rapper T. I. unloaded on black celebrities who... | 0 |
| **20796** | 20796 | N.F.L. Playoffs: Schedule, Matchups and Odds -... | Benjamin Hoffman | When the Green Bay Packers lost to the Washing... | 0 |
| **20797** | 20797 | Macy’s Is Said to Receive Takeover Approach by... | Michael J. de la Merced and Rachel Abrams | The Macy’s of today grew from the union of sev... | 0 |
| **20798** | 20798 | NATO, Russia To Hold Parallel Exercises In Bal... | Alex Ansary | NATO, Russia To Hold Parallel Exercises In Bal... | 1 |
| **20799** | 20799 | What Keeps the F-35 Alive | David Swanson | David Swanson is an author, activist, journa... | 1 |

20800 rows × 5 columns

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20800 entries, 0 to 20799
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      20800 non-null  int64 
 1   title   20242 non-null  object
 2   author  18843 non-null  object
 3   text    20761 non-null  object
 4   label   20800 non-null  int64 
dtypes: int64(2), object(3)
memory usage: 812.6+ KB
```

In [6]:
```
df.isna().sum()
```

Out[6]:
```
id           0
title      558
author    1957
text        39
label        0
dtype: int64
```

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
id           0
title      558
author    1957
text        39
label        0
dtype: int64
```

In [8]:
```
df['id'].value_counts()
```

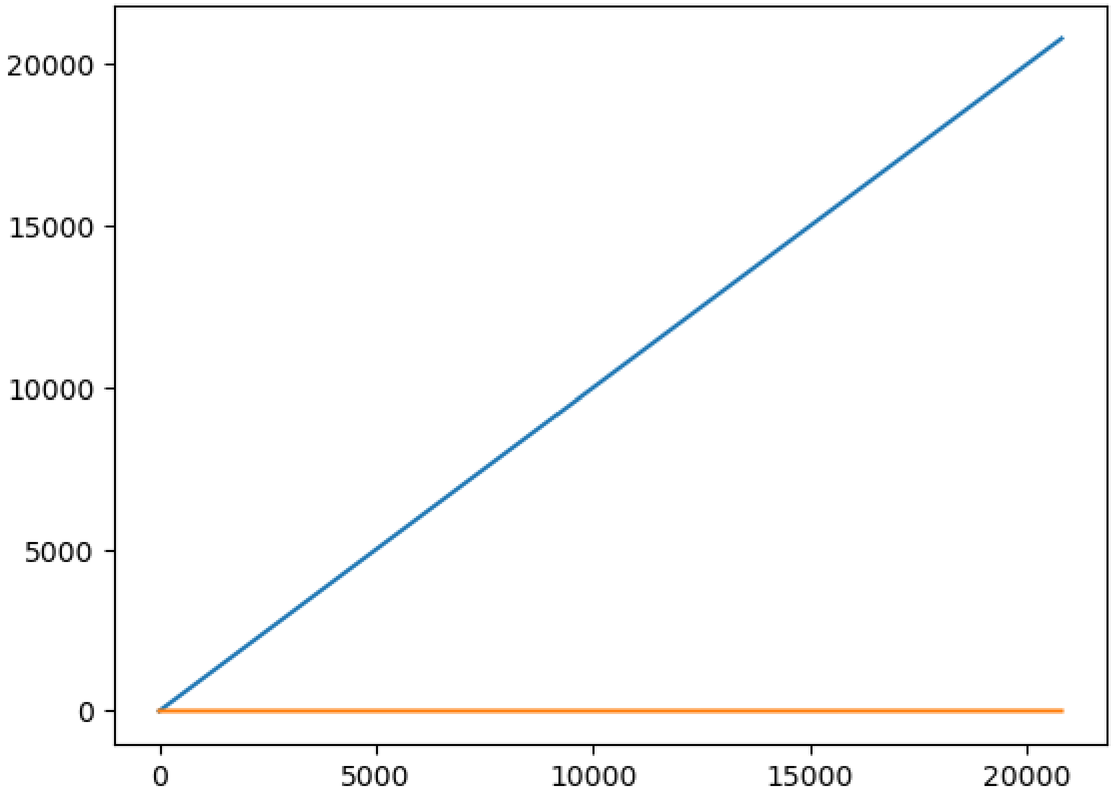
Out[8]:
```
0        1
13854    1
13872    1
13871    1
13870    1
        ..
6931     1
6930     1
6929     1
6928     1
20799    1
Name: id, Length: 20800, dtype: int64
```

In [9]:
```
df.shape
```

Out[9]:
```
(20800, 5)
```

In [10]:
```
import matplotlib.pyplot as plt
```

In [11]:
```
plt.plot(df['id'])
plt.plot(df['label']);
```



In [12]:
```python
df.describe()
```

Out[12]:

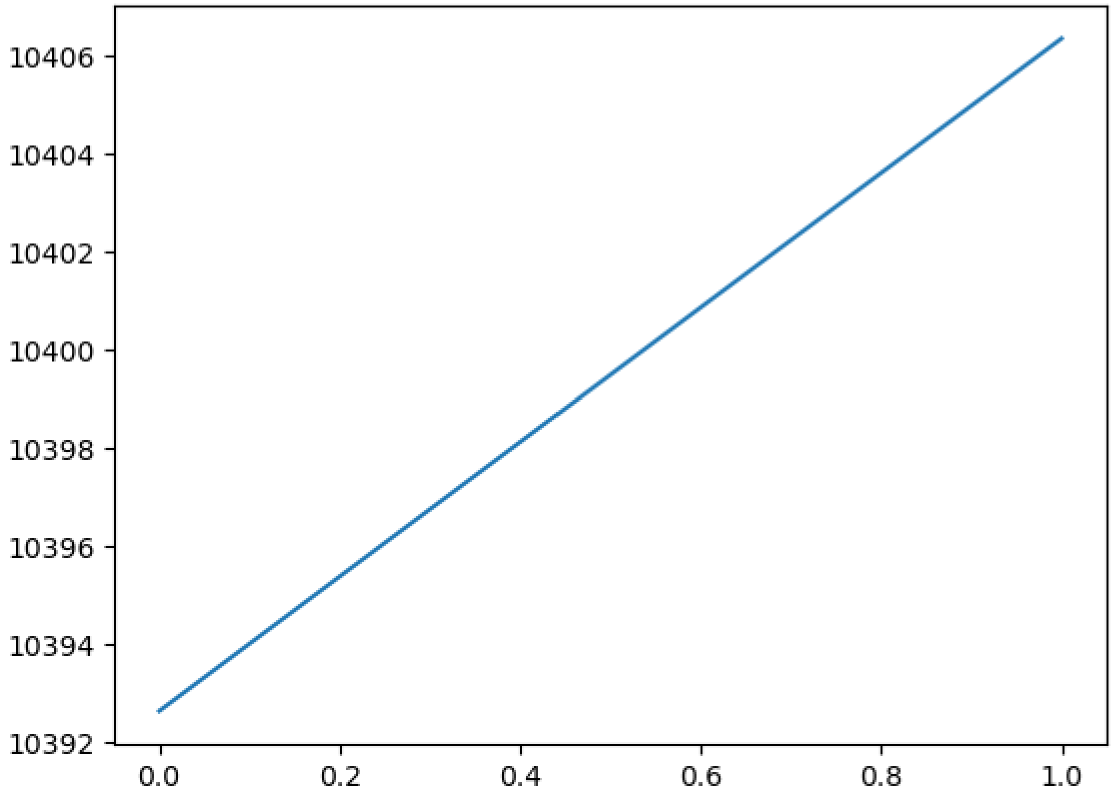
| | id | label |
|---|---|---|
| **count** | 20800.000000 | 20800.000000 |
| **mean** | 10399.500000 | 0.500625 |
| **std** | 6004.587135 | 0.500012 |
| **min** | 0.000000 | 0.000000 |
| **25%** | 5199.750000 | 0.000000 |
| **50%** | 10399.500000 | 1.000000 |
| **75%** | 15599.250000 | 1.000000 |
| **max** | 20799.000000 | 1.000000 |

In [13]:
```python
df.groupby('label').mean()
```

Out[13]:

| | id |
|---|---|
| **label** | |
| **0** | 10392.644171 |
| **1** | 10406.338711 |

In [14]:
```
plt.plot(df.groupby('label').mean());
```



## Importing some remaining dependencies for the project

In [15]:
```
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [16]:
```python
print(stopwords.words('english')), # not found, we need to download it stopwords
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "yo
u'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her',
'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'wha
t', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'wer
e', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'th
e', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'abou
t', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'fro
m', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here',
'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 's
ome', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can',
'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ai
n', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'has
n', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'need
n', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "w
on't", 'wouldn', "wouldn't"]
```

Out[16]:
```
(None,)
```

In [17]:
```python
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Error loading stopwords: <urlopen error [Errno 11001]
[nltk_data]     getaddrinfo failed>
```

Out[17]:
```
False
```

In [18]:
```
# repeat
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "yo
u'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her',
'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'wha
t', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'wer
e', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'th
e', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'abou
t', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'fro
m', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here',
'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 's
ome', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can',
'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ai
n', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'has
n', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'need
n', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "w
on't", 'wouldn', "wouldn't"]
```

## Data Pre-processing

In [19]:
```
df.head()
```

Out[19]:

| | id | title | author | text | label |
|---|---|---|---|---|---|
| **0** | 0 | House Dem Aide: We Didn’t Even See Comey’s Let... | Darrell Lucus | House Dem Aide: We Didn’t Even See Comey’s Let... | 1 |
| **1** | 1 | FLYNN: Hillary Clinton, Big Woman on Campus - ... | Daniel J. Flynn | Ever get the feeling your life circles the rou... | 0 |
| **2** | 2 | Why the Truth Might Get You Fired | Consortiumnews.com | Why the Truth Might Get You Fired October 29, ... | 1 |
| **3** | 3 | 15 Civilians Killed In Single US Airstrike Hav... | Jessica Purkiss | Videos 15 Civilians Killed In Single US Airstr... | 1 |
| **4** | 4 | Iranian woman jailed for fictional unpublished... | Howard Portnoy | Print \nAn Iranian woman has been sentenced to... | 1 |

In [20]:
```
df.isna().sum()
```

Out[20]:
```
id           0
title      558
author    1957
text        39
label        0
dtype: int64
```

In [21]:
```
# replacing the null values with empty string

df = df.fillna('')
```

In [22]:
```
# Emerging the author name and news title
df['content'] = df['author']+' '+df['title']
```

In [23]:
```
print(df['content'])
```

```
0        Darrell Lucus House Dem Aide: We Didn’t Even S...
1        Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2        Consortiumnews.com Why the Truth Might Get You...
3        Jessica Purkiss 15 Civilians Killed In Single ...
4        Howard Portnoy Iranian woman jailed for fictio...
                               ...
20795    Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796    Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797    Michael J. de la Merced and Rachel Abrams Macy...
20798    Alex Ansary NATO, Russia To Hold Parallel Exer...
20799              David Swanson What Keeps the F-35 Alive
Name: content, Length: 20800, dtype: object
```

## Separating the data and label

In [24]:
```
x = df.drop(columns='label', axis = 1)
y = df['label']
```

In [25]:
```
x, y
```

Out[25]:
```
(          id                                              title  \
0          0  House Dem Aide: We Didn’t Even See Comey’s Let...
1          1  FLYNN: Hillary Clinton, Big Woman on Campus - ...
2          2                  Why the Truth Might Get You Fired
3          3  15 Civilians Killed In Single US Airstrike Hav...
4          4  Iranian woman jailed for fictional unpublished...
...      ...                                                ...
20795  20795  Rapper T.I.: Trump a ’Poster Child For White S...
20796  20796  N.F.L. Playoffs: Schedule, Matchups and Odds -...
20797  20797  Macy’s Is Said to Receive Takeover Approach by...
20798  20798  NATO, Russia To Hold Parallel Exercises In Bal...
20799  20799                          What Keeps the F-35 Alive

                                          author  \
0                                  Darrell Lucus
1                                Daniel J. Flynn
2                             Consortiumnews.com
3                                Jessica Purkiss
4                                 Howard Portnoy
...                                          ...
20795                              Jerome Hudson
20796                           Benjamin Hoffman
20797  Michael J. de la Merced and Rachel Abrams
20798                                Alex Ansary
20799                              David Swanson

                                                    text  \
0      House Dem Aide: We Didn’t Even See Comey’s Let...
1      Ever get the feeling your life circles the rou...
2      Why the Truth Might Get You Fired October 29, ...
3      Videos 15 Civilians Killed In Single US Airstr...
4      Print \nAn Iranian woman has been sentenced to...
...                                                  ...
20795  Rapper T. I. unloaded on black celebrities who...
20796  When the Green Bay Packers lost to the Washing...
20797  The Macy’s of today grew from the union of sev...
20798  NATO, Russia To Hold Parallel Exercises In Bal...
20799    David Swanson is an author, activist, journa...

                                                 content
0      Darrell Lucus House Dem Aide: We Didn’t Even S...
1      Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2      Consortiumnews.com Why the Truth Might Get You...
3      Jessica Purkiss 15 Civilians Killed In Single ...
4      Howard Portnoy Iranian woman jailed for fictio...
...                                                  ...
20795  Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796  Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797  Michael J. de la Merced and Rachel Abrams Macy...
20798  Alex Ansary NATO, Russia To Hold Parallel Exer...
20799            David Swanson What Keeps the F-35 Alive

[20800 rows x 5 columns],
0        1
1        0
2        1
3        1
4        1
        ..
20795    0
20796    0
20797    0
20798    1
20799    1
Name: label, Length: 20800, dtype: int64)
```

## Stemming

Stemming is the process of reducing a word to its root word.

Example: actor, actress, acting ---> act(root word)

In [26]:
```python
# creating a function

import re
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords

def stemming(content):
    # Initialize the PorterStemmer object
    port_stem = PorterStemmer()

    # Remove non-alphabetic characters
    stemmed_content = re.sub('[^a-zA-Z]', ' ', content)

    # Convert to lowercase
    stemmed_content = stemmed_content.lower()

    # Split into individual words
    stemmed_content = stemmed_content.split()

    # Perform stemming and remove stopwords
    stemmed_content = [port_stem.stem(word) for word in stemmed_content if word not in stopwords.word

    # Join stemmed words into a single string
    stemmed_content = ' '.join(stemmed_content)

    return stemmed_content

```

The given code defines a function called `stemming` that takes a parameter `content`. The purpose of this function is to perform stemming on the input content.

Stemming is a process in natural language processing that reduces words to their base or root form. It helps in standardizing different forms of a word to a common base form, which can be beneficial for tasks like text analysis, information retrieval, and language processing.

Here's a step-by-step explanation of the code:

1. `stemmed_content = re.sub('[^a-zA-Z]', ' ', content)`: This line uses the `re.sub()` function from the `re` module to substitute any character that is not a letter (specified using the regular expression `[^a-zA-Z]`) in the `content` variable with a space (' '). This effectively removes any non-alphabetic characters from the content.

2. `stemmed_content = stemmed_content.lower()` : This line converts all the alphabetic characters in the `stemmed_content` variable to lowercase. This step is often performed to ensure case insensitivity during further processing.
3. `stemmed_content = stemmed_content.split()` : This line splits the `stemmed_content` string into a list of words. The `split()` method is called without any arguments, which means it will split the string at whitespace characters (e.g., spaces, tabs, newlines) and return a list of individual words.
4. `stemmed_content = [port_stem.stem(word) for word in stemmed_content if not word in stopwords.words('english')]` : This line uses a list comprehension to iterate over each word in the `stemmed_content` list. For each word, it checks if the word is not in the set of English stopwords (common words like "the," "is," "and," etc. that are often removed from text for analysis purposes). If the word is not a stopword, it applies stemming using a stemming algorithm represented by `port_stem.stem(word)` . The result is a list of stemmed words.
5. `stemmed_content = ' '.join(stemmed_content)` : This line joins the stemmed words in the `stemmed_content` list back into a single string, separated by a space (' '). This step is performed to obtain the final stemmed content as a string.
6. Finally, the function returns the `stemmed_content` string as the output.

To use this function, you would need to import the necessary modules ( `re` , `nltk.stem.porter` , and `nltk.corpus.stopwords` ) and have the NLTK library installed. Additionally, the `port_stem` object needs to be initialized as an

In [27]:
```
df['content'] = df['content'].apply(stemming)
```

In [28]:
```
df['content']
```

Out[28]:
```
0        darrel lucu hous dem aid even see comey letter...
1        daniel j flynn flynn hillari clinton big woman...
2                   consortiumnew com truth might get fire
3        jessica purkiss civilian kill singl us airstri...
4        howard portnoy iranian woman jail fiction unpu...
                               ...                        
20795    jerom hudson rapper trump poster child white s...
20796    benjamin hoffman n f l playoff schedul matchup...
20797    michael j de la merc rachel abram maci said re...
20798    alex ansari nato russia hold parallel exercis ...
20799                            david swanson keep f aliv
Name: content, Length: 20800, dtype: object
```

In [30]:
```
# Separating the data and label
x = df['content'].values
y = df['label'].values
```

In [36]:
```
x, x.shape
```

Out[36]:
```
(array(['darrel lucu hous dem aid even see comey letter jason chaffetz tweet',
        'daniel j flynn flynn hillari clinton big woman campu breitbart',
        'consortiumnew com truth might get fire', ...,
        'michael j de la merc rachel abram maci said receiv takeov approach hudson bay new york time',
        'alex ansari nato russia hold parallel exercis balkan',
        'david swanson keep f aliv'], dtype=object),
 (20800,))
```

In [35]:
```
y.shape, y
```

Out[35]:
```
((20800,), array([1, 0, 1, ..., 0, 1, 1], dtype=int64))
```

In [37]:
```
# converting the textual data to numerical data

vectorizer = TfidfVectorizer()
vectorizer.fit(x)
x = vectorizer.transform(x)
```

This code snippet is using the TfidfVectorizer class from the scikit-learn library to convert a collection of raw text documents into a numerical feature matrix. Here's what each line does:

1. `vectorizer = TfidfVectorizer()` : This creates an instance of the TfidfVectorizer class, which is used for feature extraction from text data using the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm.
2. `vectorizer.fit(x)` : This line fits the vectorizer to the given input data `x` . In other words, it analyzes the text documents in `x` to learn the vocabulary and document frequencies.
3. `x = vectorizer.transform(x)` : This line transforms the input data `x` into a sparse matrix representation using the vocabulary and document frequencies learned by the vectorizer. Each row of the resulting matrix represents a document, and each column represents a unique word in the vocabulary. The values in the matrix correspond to the TF-IDF scores of the words in the documents.

Overall, this code snippet performs text vectorization using the TF-IDF algorithm, converting a collection of text documents into a numerical representation that can be used for machine learning tasks such as text classification or clustering.

In [41]:
```
print(x)
```

```
  (0, 15686)	0.28485063562728646
  (0, 13473)	0.2565896679337957
  (0, 8909)	0.3635963806326075
  (0, 8630)	0.29212514087043684
  (0, 7692)	0.24785219520671603
  (0, 7005)	0.21874169089359144
  (0, 4973)	0.233316966909351
  (0, 3792)	0.2705332480845492
  (0, 3600)	0.3598939188262559
  (0, 2959)	0.2468450128533713
  (0, 2483)	0.3676519686797209
  (0, 267)	0.27010124977708766
  (1, 16799)	0.30071745655510157
  (1, 6816)	0.1904660198296849
  (1, 5503)	0.7143299355715573
  (1, 3568)	0.26373768806048464
  (1, 2813)	0.19094574062359204
  (1, 2223)	0.3827320386859759
  (1, 1894)	0.15521974226349364
  (1, 1497)	0.2939891562094648
  (2, 15611)	0.41544962664721613
  (2, 9620)	0.49351492943649944
  (2, 5968)	0.3474613386728292
  (2, 5389)	0.3866530551182615
  (2, 3103)	0.46097489583229645
  :	:
  (20797, 13122)	0.2482526352197606
  (20797, 12344)	0.27263457663336677
  (20797, 12138)	0.24778257724396507
  (20797, 10306)	0.08038079000566466
  (20797, 9588)	0.174553480255222
  (20797, 9518)	0.2954204003420313
  (20797, 8988)	0.36160868928090795
  (20797, 8364)	0.22322585870464118
  (20797, 7042)	0.21799048897828688
  (20797, 3643)	0.21155500613623743
  (20797, 1287)	0.33538056804139865
  (20797, 699)	0.30685846079762347
  (20797, 43)	0.29710241860700626
  (20798, 13046)	0.22363267488270608
  (20798, 11052)	0.4460515589182236
  (20798, 10177)	0.3192496370187028
  (20798, 6889)	0.32496285694299426
  (20798, 5032)	0.4083701450239529
  (20798, 1125)	0.4460515589182236
  (20798, 588)	0.3112141524638974
  (20798, 350)	0.28446937819072576
  (20799, 14852)	0.5677577267055112
  (20799, 8036)	0.45983893273780013
  (20799, 3623)	0.37927626273066584
  (20799, 377)	0.5677577267055112
```

In [43]:

```
x.shape
```

Out[43]:

```
(20800, 17128)
```

## Splitting the dataset to training and test data

In [44]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, stratify=y, random_state=42)
```

In [50]:

```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

Out[50]:

```
((16640, 17128), (4160, 17128), (16640,), (4160,))
```

## Training the model

In [51]:

```
model = LogisticRegression()
```

In [52]:

```
model.fit(x_train, y_train)
```

Out[52]:

```
LogisticRegression()
```

In [56]:

```
y_preds = model.predict(x_test)
y_preds
```

Out[56]:

```
array([0, 0, 1, ..., 0, 1, 0], dtype=int64)
```

In [60]:
```
accuracy_score(y_test, y_preds)
```

Out[60]:
```
0.9752403846153846
```

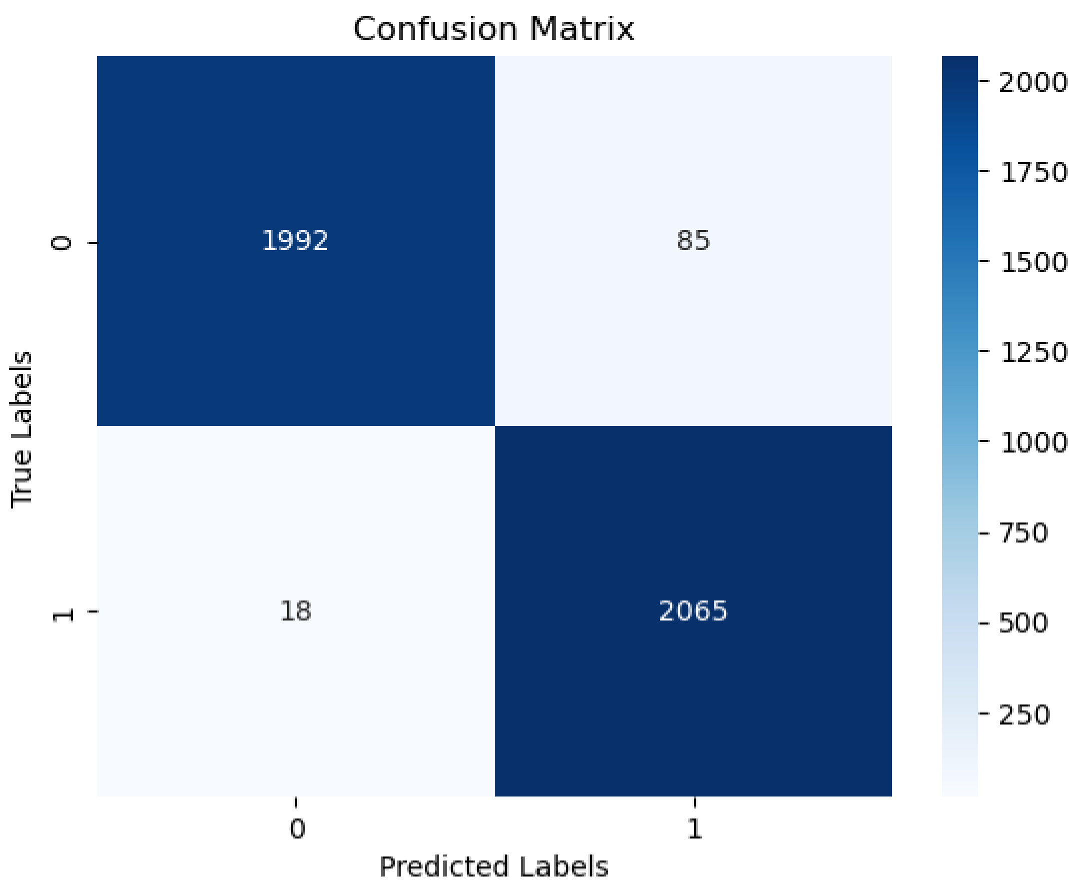
In [57]:
```
from sklearn.metrics import confusion_matrix

# Assuming you have the true labels for the test dataset in y_true
confusion = confusion_matrix(y_test, y_preds)
print(confusion)
```

```
[[1992   85]
 [  18 2065]]
```

In [58]:
```
# Create a DataFrame with the actual and predicted values
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_preds})

# Print the DataFrame
print(df)
```

```
      Actual  Predicted
0          0          0
1          0          0
2          1          1
3          1          1
4          0          0
...      ...        ...
4155       1          1
4156       0          0
4157       0          0
4158       1          1
4159       0          0

[4160 rows x 2 columns]
```

In [59]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming you have the confusion matrix stored in the 'confusion' variable

# Create a heatmap using seaborn and format annotations as integers
sns.heatmap(confusion, annot=True, fmt='d', cmap='Blues')

# Add labels, title, and axis ticks
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')

# Show the plot
plt.show()

```


Confusion Matrix
1992
85
18
2065
True Labels
Predicted Labels
0
1
2000
1750
1500
1250
1000
750
500
250

## Evaluation

In [61]:
```python
from sklearn.metrics import classification_report

class_report = classification_report(y_test, y_preds)
print("Classification Report:")
print(class_report)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.96      0.97      2077
           1       0.96      0.99      0.98      2083

    accuracy                           0.98      4160
   macro avg       0.98      0.98      0.98      4160
weighted avg       0.98      0.98      0.98      4160
```

## Making a predictive system

In [68]:
```python
x_new = x_test[0]

prediction = model.predict(x_new)
prediction

if prediction[0]== 0:
    print('The news is real')
else:
    print('The news is Fake')
```

```
The news is real
```

In [67]:
```python
print(y_test[0])
```

```
0
```

We will be using the full data in the next project and making prediction with the provided test dataset. That means the training data will be splitted into training and validation data and will be used to make prediction on the new test dataset

In [ ]:

```

```